ESTRUCTURA DE DATOS 1
Código ST0245

Taller en Sala 8
Pilas y Colas

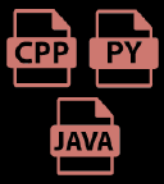
Objetivo: **1.** Solucionar problemas del mundo real con algoritmos. **2.** Escoger la estructura de datos adecuada para resolver un problema dado. **3.** Escribir programas que utilicen las estructuras de datos listas, pilas y colas.

Consideraciones: Lean y verifiquen las consideraciones de entrega,

Trabajo en Parejas

Mañana, plazo de entrega

Docente entrega plantilla de código en GitHub



Sí .cpp, .py o .java

No .zip, .txt, html o .doc

Alumnos entregan código sin comprimir GitHub

En la carpeta Github del curso, hay un código iniciado y un código de pruebas (tests) que pueden explorar para solucionar los ejercicios

Estructura del documento: **a)** Datos de *vida real*, **b)** I*ntroducción* a un problema, **c)** Problema a resolver, **d)** Ayudas. Identifiquen esos elementos así:

a)

b)

c)

d)

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Ejercicios a resolver

En la vida real, las pilas se utilizan para representar los llamados a funciones en los lenguajes de programación. Si la pila se llena, ocurre un error muy común llamado *Stack Overflow* http://bit.ly/2wbiNgH.

Las colas se utilizan para implementar planificadores de procesos, de disco duro, de tarjetas de sonido y de dispositivos de entrada y salida http://bit.ly/2xotgH6

1 La notación polaca inversa ha sido utilizada desde la creación del computador porque representa la forma en que se evalúan las operaciones aritméticas en los procesadores. Por esta razón, se utiliza mucho en lenguajes formales y compiladores.

Adicionalmente, durante los años 70s y 80s, *Hewlett Packard* desarrolló varias calculadoras científicas, como *la HP48GX*, que utilizaban esta notación y guardaban los subtotales de las operaciones en una pila.

Escriban un algoritmo para evaluar expresiones en notación polaca inversa utilizando pilas. Los números y símbolos se separan con espacios. Como un ejemplo, si el algoritmo recibe la cadena de caracteres “3 5 * 2 +”, debe retornar 17. Si el algoritmo recibe la cadena de caracteres “6 5 – 4 +”, debe retornar 5.

2 Se tiene un almacén donde se encuentran las neveras fabricadas por una planta:

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

- Las primeras neveras que fueron fabricadas están de últimas dentro del almacén; las últimas neveras fabricadas recientemente, aparecen más cerca de la puerta del almacén.
- Para sacar una nevera que está atrás, primero habría que quitar la que está adelante.
- Los datos de cada nevera son el código (representado por un número) y la descripción (representada por un texto).
- El almacén dispone de una sola puerta por donde las neveras entran las neveras para ser almacenadas y por donde salen las neveras que se van a distribuir a las tiendas.
- Se tiene una lista de solicitudes de neveras, realizadas por las tiendas, donde aparece el nombre de la tienda y la cantidad solicitada de neveras. Se deben atender primero las solicitudes más antiguas y luego las más recientes.

Teniendo en cuenta lo anterior, implementen un algoritmo que reciba las neveras, en la forma en que están ordenadas en el almacén, y las solicitudes, según el orden en que llegaron, y que imprima qué neveras del almacén quedan asignadas a cada tienda.

3 **[Ejercicio Opcional]** La siguiente pila contiene los elementos 1, 2, 3…

```
Stack<Integer> stack = new Stack<Integer>();
stack.push(1);
stack.push(2);
stack.push(3);…
```

La pila tiene los elementos en orden inverso, es decir, …3, 2, 1. **Escriban un código que invierta sus elementos de orden.**

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

4 **[Ejercicio Opcional]** Un cristiano, un musulmán y un judío llegan a un bar. El cristiano llega primero, el musulmán llega segundo y el judío llega de tercero.

```
Queue<String> queue = new LinkedList<String>();
queue.add("Juan");
queue.add("María");
queue.add("Pedro");
```

Atiéndalos, por favor, en el orden en que llegaron, imprimiendo "*Atendiendo a*" y el nombre de la persona.

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

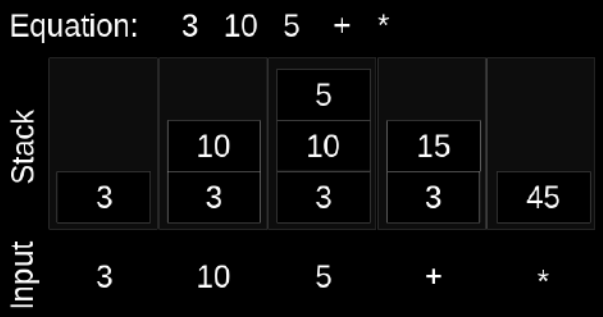
Ayudas para resolver los Ejercicios

Ejercicio 1

Pista 1: Usen el método `String.split(" ")` para dividir una cadena por espacios

Pista 2: Utilicen Pilas

Ejemplo 1: Este es un ejemplo cómo funciona el algoritmo



Pista 3: Utilicen el método `Character.isDigit(c)` para saber si un caracter es dígito

Pista 4: Utilicen el método Character.getNumericValue(c) para saber el valor numérico de un caracter

Pista 5: Utilicen el método `Integer.ParseInt(S)` para convertir una cadena en entero.

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Ejercicio 2

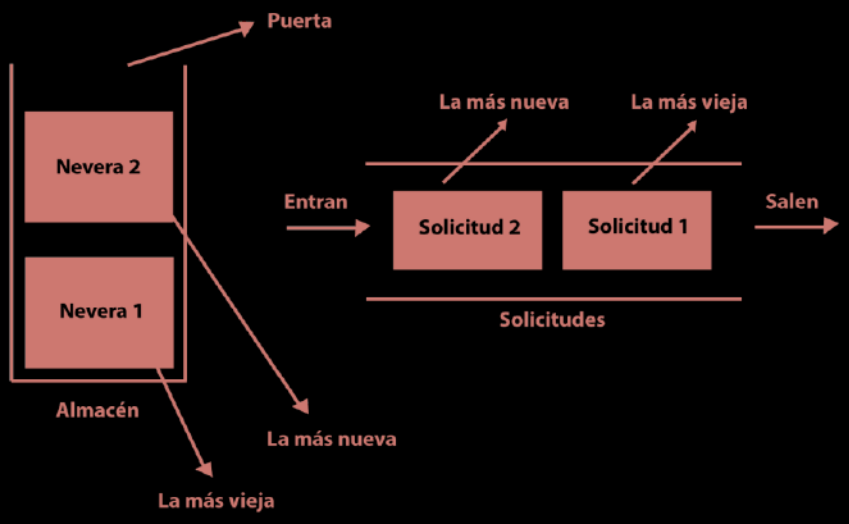
Pista 1: Escriban un método como el siguiente:

```
public static void ejercicio2( ??? neveras, ??? solicitudes)
```

Nota 1: Este algoritmo se puede hacer complejidad *O(n.m)* donde *n* es el número de solicitudes y *m* el máximo número de neveras en una solicitud.

Pista 2: Utilicen listas enlazadas

Pista 3: No supongan, erróneamente, que hay suficientes neveras para cumplir las solicitudes



PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Ejemplo 1: Para las siguientes neveras y solicitudes,

```
almacen = [(1,"haceb"), (2,"lg"), (3,"ibm"), (4,"haceb"),
(5,"lg"), (6,"ibm"),(7,"haceb"), (8,"lg"), (9,"ibm"),(8,"lg"),
(9,"ibm")]

solicitudes = [("eafit", 10), ("la14", 2), ("olimpica", 4),
("éxito", 1)]
```

Donde "*éxito*" fue la primera solicitud y "*eafit*" la última, e "*ibm*" con código 9 fue la última nevera ingresada a la bodega, la respuesta debe ser:

```
[('exito', [(9, 'ibm')]),
('olimpica', [(8, 'lg'), (9, 'ibm'), (8, 'lg'), (7, 'haceb')]),
('la14', [(6, 'ibm'), (5, 'lg')]),
('eafit', [(4, 'haceb'), (3, 'ibm'), (2, 'lg'), (1, 'haceb')])]
```

Ejercicio 3 y 4

Pista 1: Utilicen funciones de parámetro variable para poder recibir los elementos que van a agregar a la pila o cola, de esta forma:

```
static void fun(int ...a)
{
    for(int i : a)
            System.out.print(i + " ");
}
```

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Error Común 1: Un error común es escribir un ciclo de esta forma

```
for(int i = 0; i < pila.size(); i++)
  pila.pop()
```

El problema es que el tamaño de la pila cambia cada vez que se elimina un elemento de la pila. Lo mismo sucede con las colas. La solución es usar un ciclo *while*:

```
while(pila.size() != 0)
  pila.pop()
```

Error Común 2:

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

¿Alguna inquietud?

CONTACTO

Docente Mauricio Toro Bermúdez
Teléfono: (+57) (4) 261 95 00 Ext. 9473
Correo: mtorobe@eafit.edu.co
Oficina: 19- 627

Agenden una cita dando clic en la pestaña -*Semana*- de http://bit.ly/2gzVg10